LatticeCORE

2.5 Gbps Ethernet PCS IP Core User's Guide

March 2012
IPUG99_01.0

Table of Contents

Introduction

The Lattice 2.5 Gbps Ethernet PCS IP core implements the state machine functions for the physical coding sub-layer (PCS) described in the IEEE 802.3z (1000BaseX) specification. Note that the IEEE specification describes a PCS that operates at 1Gbps. Therefore, this 2.5G PCS IP core is non-standard with respect to the IEEE specification. The two major non-compliances are data rate (2.5 Gbps instead of 1 Gbps) and GMII data bus width (16 bits instead of 8 bits).

This PCS IP core was specifically developed to operate with the Lattice 2.5 Gbps MAC IP core. The Lattice 2.5G PCS and MAC IP cores are 100% compatible and can be used to create a full PHY/MAC Ethernet data path that operates at 2.5 Gbps.

This document describes the 2.5 Gbps Ethernet PCS IP core's operation, and provides instructions for generating the core through the Lattice IPexpress™ tool, and for instantiating, synthesizing, and simulating the core.

Quick Facts

Table 1-1 gives quick facts about the 2.5 Gbps Ethernet PCS IP core.

Table 1-1. 2.5 Gbps Ethernet PCS IP Core Quick Facts

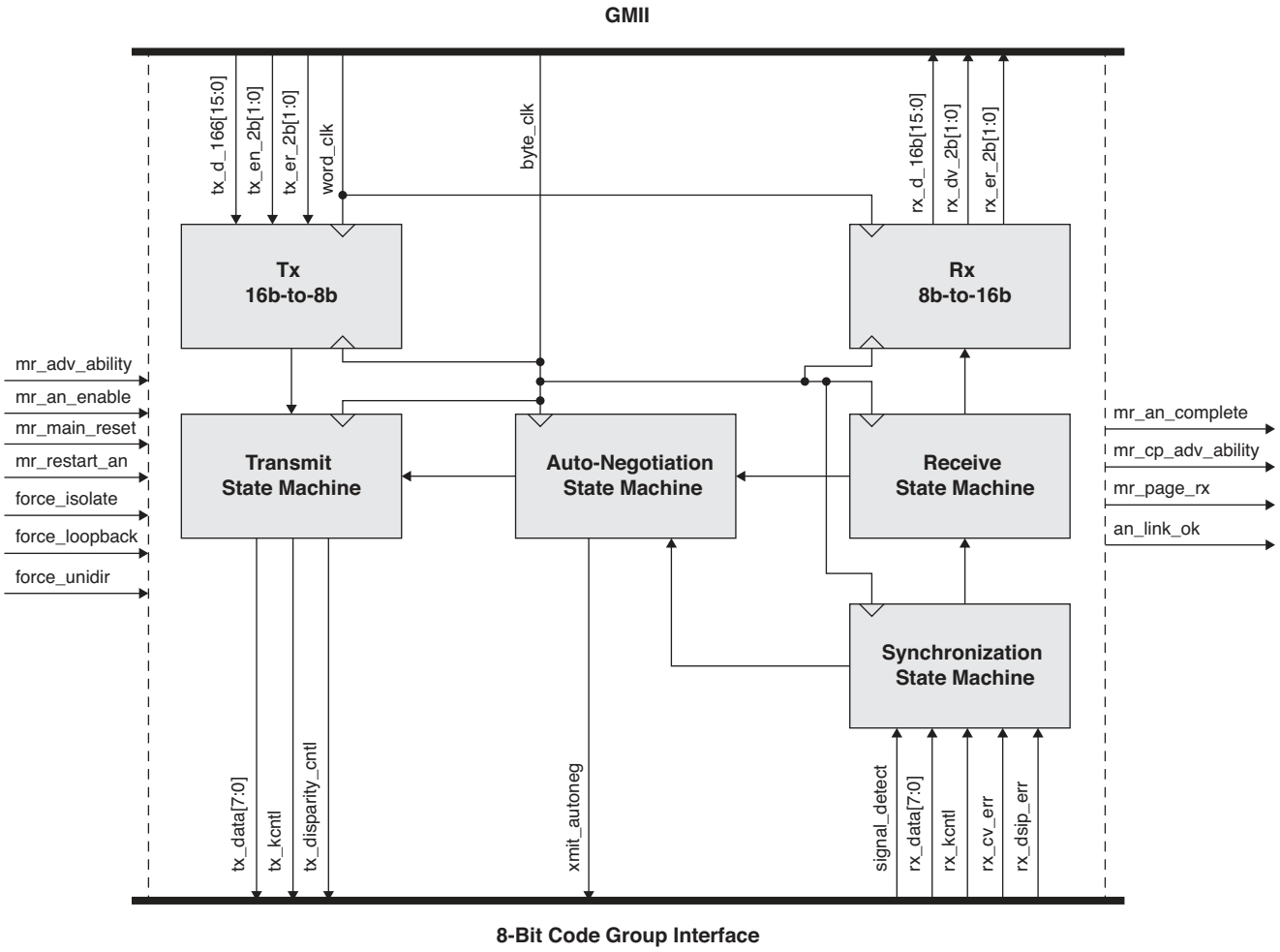
		IP Configuration
Core Requirements	FPGA Families Supported	LatticeECP3™
	Minimal Device Needed	LFE3-17EA-8FTN256C
Resource Utilization	Data Path Width	16
	LUTs	600
	sysMEM™ EBRs	0
	Registers	900
Design Tool Support	Lattice Implementation	Lattice Diamond® 1.3
	Synthesis	Synopsys® Synplify® Pro for Lattice E-2011.03L
		Mentor Graphics® Precision® RTL
	Simulation	Aldec® Active-HDL® 8.2 Lattice Edition
		Mentor Graphics ModelSim® SE (Verilog only)

Features

- Implements the transmit, receive, and auto-negotiation functions of the IEEE 802.3z specification
- 16-bit GMII interface operating at 156.25 MHz (2.5 Gbps)
- 8-bit code-group interface operating at 312.5 MHz (2.5 Gbps)
- Parallel signal interface for control and status management

The 2.5 Gbps Ethernet PCS IP core converts GMII frames into 8-bit code groups in both transmit and receive directions and performs auto-negotiation with a link partner as described in IEEE 802.3z specification. The IP core's block diagram is shown in Figure 2-1.

Figure 2-1. 2.5 Gbps Ethernet PCS IP Core Block Diagram



As mentioned earlier, the 2.5 Gbps Ethernet PCS IP core is intended to be combined with the Lattice 2.5G MAC IP core to form a complete 2.5G MAC/PHY Ethernet data path. Figure 2-2. below shows how these two IP cores can be combined.

Figure 2-2. 2.5 Gbps MAC/PHY Ethernet Reference Design

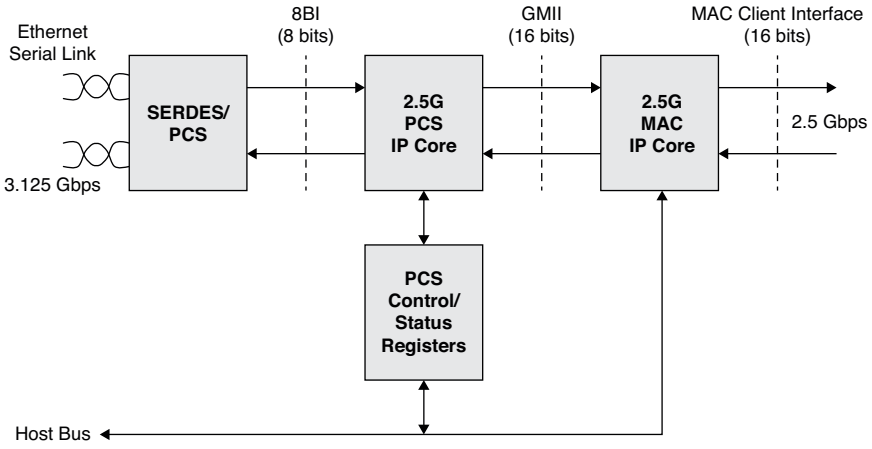
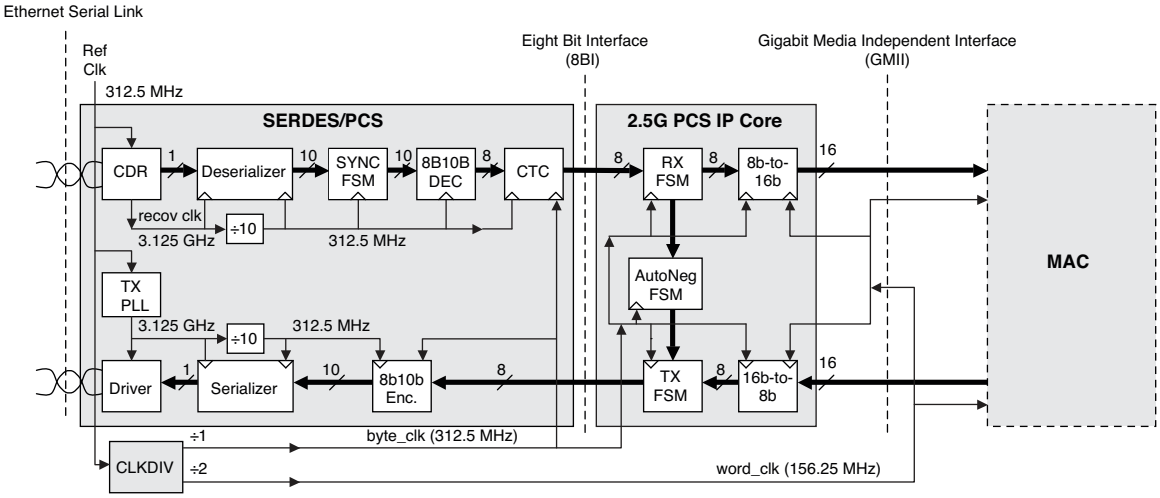


Figure 2-3 shows typical clock network connections between the 2.5 Gbps Ethernet PCS IP core, the SERDES, and a MAC.

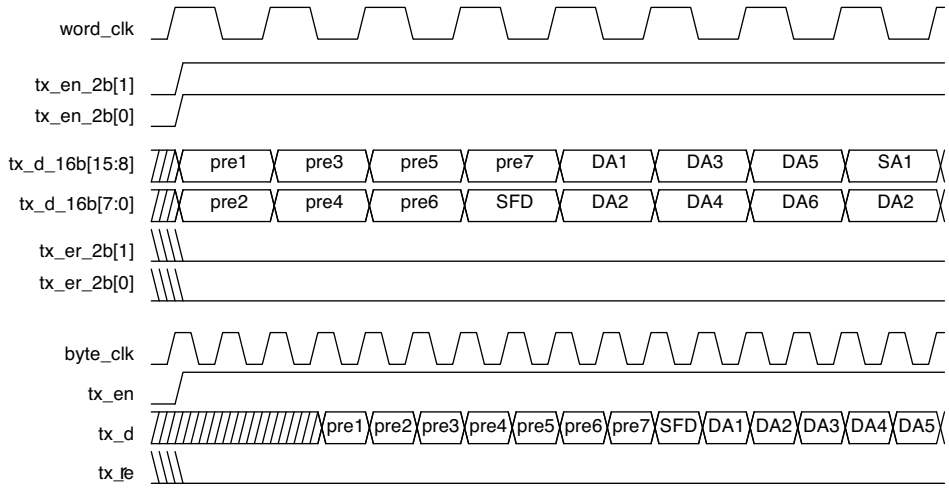
Figure 2-3. Typical Clocking Network



Transmit 16-bit to 8-bit Conversion

This circuit block converts the double-byte wide Tx GMII bus into a single-byte wide Tx GMII bus. The 16-bit bus operates at 156.25 Mhz. The 8-bit bus operates at 312.5 Mhz. Figure 2-4 shows typical bus timing.

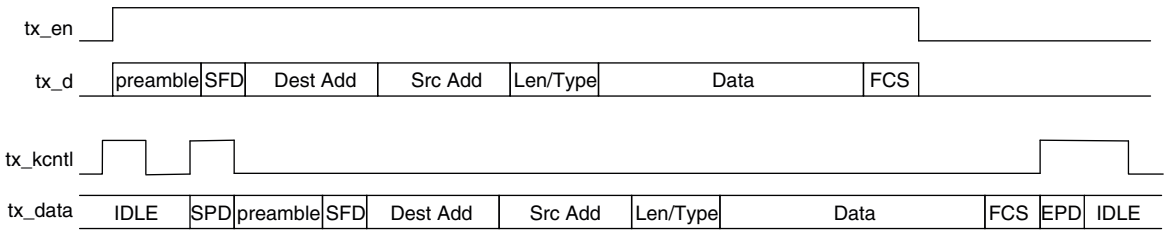
Figure 2-4. Timing Diagram for Tx 16b-to-8b Circuit Block



Transmit State Machine

The transmit state machine implements the transmit functions described in clause 36 of the IEEE802.3 specification. The state machine's main purpose is to convert GMII data frames into code groups. A typical timing diagram for this circuit block is shown in Figure 2-5. Note that the state machine in this IP core does not fully implement the conversion to 10-bit code groups as specified in the IEEE802.3 specification. Instead, partial conversion to 8-bit code groups is performed. A separate encoder in the Lattice SERDES completes the full conversion to 10-bit code groups.

Figure 2-5. Typical Transmit Timing Diagram



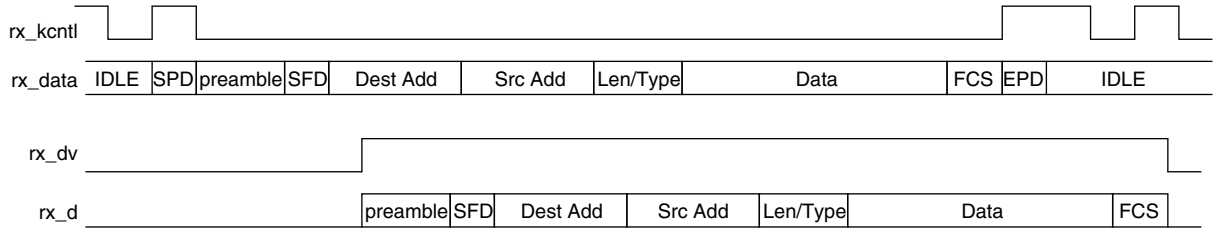
Synchronization State Machine

The synchronization state machine implements the alignment functions described in clause 36 of the IEEE802.3 specification. The state machine's main purpose is to determine whether incoming code groups are properly aligned. Once alignment is attained, proper code groups are passed to the receive state machine. If alignment is lost for an extended period, an auto-negotiation restart is triggered.

Receive State Machine

The receive state machine implements the receive functions described in clause 36 of the IEEE802.3 specification. The state machine's main purpose is to convert code groups into GMII data frames. A typical timing diagram for this circuit block is shown in Figure 2-6. Note that the state machine in this IP core does not fully implement the conversion from 10-bit code groups as specified in the IEEE802.3 specification. Instead, partial conversion from 8-bit code groups is performed. A separate decoder in the Lattice SERDES performs 10-bit to 8-bit code group conversions.

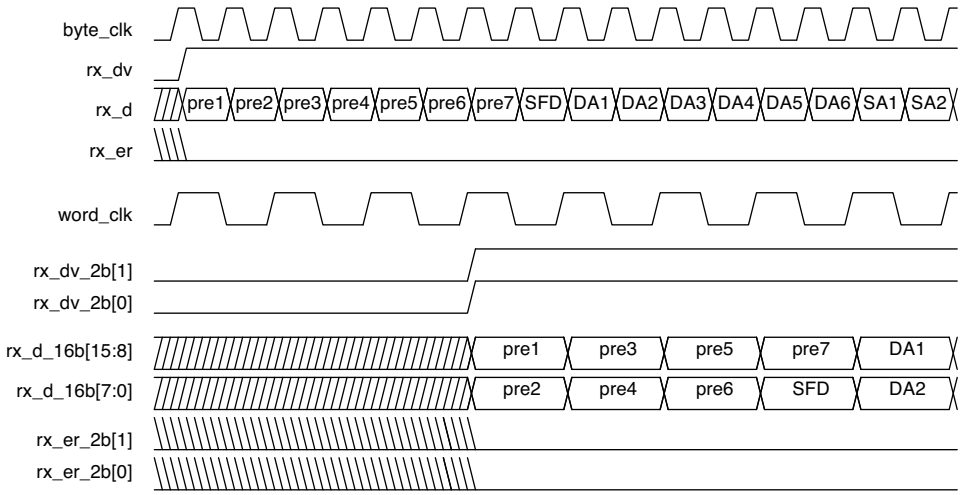
Figure 2-6. Typical Receive Timing Diagram



Receive 8-bit to 16-bit Conversion

This circuit block converts a single-byte wide Rx GMII bus into a double-byte wide Rx GMII bus. The 8-bit bus operates at 312.5 MHz. The 16-bit bus operates at 156.25 MHz. Figure 2-7 shows typical bus timing.

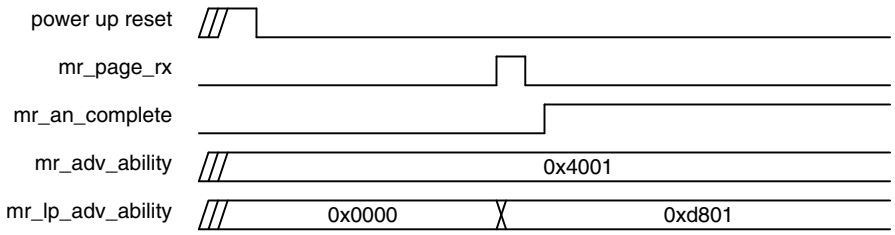
Figure 2-7. Timing Diagram for Rx 8b-to-16b Circuit Block



Auto-Negotiation State Machine

The auto-negotiation state machine implements the link configuration functions described in clause 37 of IEEE802.3 specification. Please consult the IEEE specification for a detailed description of auto-negotiation operation. From a high-level view, the primary auto-negotiation functions are: testing the physical link for proper operation, and passing link configuration information between Ethernet PHYs sitting on both sides of the link.

Figure 2-8. Typical Auto-Negotiation Timing Diagram for MAC-Side Entity



Data Path Latency Specifications

Latency is the time delay taken to propagate signals between two points in the data path. This section provides latency specifications for the transmit and receive data paths of the 2.5 Gbps Ethernet PCS IP core. The latency values are found by measuring the delay time between the leading edge of the SFD symbol of an Ethernet frame, as it passes through the data path. For example, to specify the latency of the Rx state machine, a continuous burst of constant-width Ethernet frames, with constant 12-clock-cycle inter frame gaps are applied to the head-end of the

Rx data path. Then the latency is found by measuring the difference between the leading edge of the SFD symbol into and out-of the Rx state machine. The latency specifications are given in units of byte-clock-cycles (312.5 MHz clock).

Table 2-1 shows the latency of the transmit data path.

Table 2-1. Transmit Data Path Latency Specifications

Tx 16b-to-8b Latency	Tx Pipeline Stage 1 Latency	Tx FSM Latency	Tx Pipeline Stage 2 Latency	Tx Total Latency
3	1	7	2	13

Table 2-2 shows the latency of the receive data path.

Table 2-2. Receive Data Path Latency Specifications

Rx Pipeline Stage 1 Latency	Rx FSM Latency	Rx 8b-to-16b Latency	Rx Pipeline Stage 2 Latency	Rx Total Latency
2	6	6	1	15

Signal Descriptions

Table 2-3 shows a detailed listing of all the 2.5 Gbps Ethernet PCS IP core I/O signals.

Table 2-3. 2.5 Gbps Ethernet PCS IP Core Input and Output Signals

Signal Name	I/O	Signal Description
Clock Signals		
byte_clk	In	**Byte Clock** - 312.5 MHz clock for the PCS state machines and Tx/Rx 8-bit code group interface that interconnects to the SERDES. Incoming data is sampled on rising edge of this clock. Outgoing data is launched on the rising edge of this clock.
word_clk	In	**Word Clock** - 156.25 MHz clock for the 16-bit GMII ports. Incoming data is sampled on the rising edge of this clock. Outgoing data is launched on the rising edge of this clock.
GMII Signals		
tx_d_16b[15:0]	In	**Transmit Data** - Incoming 16-bit GMII data. When the data frame reaches the Ethernet physical layer, the upper byte (bits 15:8) is sent first. Also, for each byte, the least significant bit is sent first (bit D8 for the upper byte; bit D0 for the lower byte).
tx_en_2b[1:0]	In	**Transmit Enable** - 2-bit active-high signal that asserts when incoming GMII data is valid. tx_en_2b[1] is associated with the upper byte of the GMII data (bits 15:8). tx_en_2b[0] is associated with the lower byte of the GMII data (bits 7:0). Note that when an Ethernet frame is being transmitted, most of the time both transmit enable bits are asserted simultaneously. However, at the beginning and end of the frame, it is possible for only one of the two GMII data bytes to be valid. A GMII data frame can begin and end on any byte (upper or lower).
tx_er_2b[1:0]	In	**Transmit Error** - 2-bit, active-high signal that denotes transmission errors or carrier extention events on the GMII Tx data port. tx_er_2b[1] is associated with the upper byte of the GMII data. tx_er_2b[0] is associated with the lower byte of the GMII data.
rx_d_16b[15:0]	Out	**Receive Data** - Outgoing 16-bit GMII data. When the data frame arrives at the Ethernet physical layer, the upper byte (bits 15:8) arrives first. Also, for each byte, the least significant bit arrives first (bit D8 for the upper byte; bit D0 for the lower byte).
rx_dv_2b[1:0]	Out	**Receive Data Valid** - 2-bit, active-high signal that asserts when outgoing GMII data is valid. rx_dv_2b[1] is associated with the upper byte of the GMII data (bits 15:8). rx_dv_2b[0] is associated with the lower byte of the GMII data (bits 7:0). Note that when an Ethernet frame is being received, most of the time both receive valid bits are asserted simultaneously. However, at the beginning and end of the frame, it is possible for only one of the two GMII data bytes to be valid. A GMII data frame can begin and end on any byte (upper or lower).

Table 2-3. 2.5 Gbps Ethernet PCS IP Core Input and Output Signals (Continued)

Signal Name	I/O	Signal Description
rx_er_2b[1:0]	Out	**Receive Error** - 2-bit, active-high signal that denotes transmission errors or carrier extention events on the GMII Rx data port. rx_er_2b[1] is associated with the upper byte of the GMII data. rx_er_2b[0] is associated with the lower byte of the GMII data.
8-Bit Code Group Signals		
tx_data[7:0]	Out	**8b Transmit Data** - 8-bit code group data after passing through transmit state machine.
tx_kcntl	Out	**8b Transmit K Control** - Denotes whether current code group is data or control. 1 = control, 0 = data.
tx_disparity_cntl	Out	**8b Transmit Disparity Control** - Used to influence the disparity state of an idle code group at the beginning of an inter-packet gap. 1 = Insert /I1/ if inter-packet gap begins with positive disparity, otherwise insert /I2/; 0 = normal, insert /I2/.
rx_data[7:0]	In	**8b Receive Data** - 8-bit code group data presented to the receive state machine.
rx_kcntl	In	**8b Receive K Control** - Denotes whether current code group is data or control. 1 = control, 0 = data.
rx_cv_err	In	**Rx Coding Violation Error** - Active-high signal denoting a coding violation error in the receive data path.
rx_disp_err	In	**Rx Disparity Error** - Active-high signal denoting a disparity error in the receive data path.
signal_detect	In	**Signal Detect** - Denotes status of SERDES Rx physical link. 1 = signal is good; 0 = loss of receive signal.
xmit_autoneg	Out	**Auto-negotiation XMIT Status** – This signal should be tied to the “xmit_chn” pin of the SERDES. When the signal is high, it forces the Rx data path of the SERDES to periodically insert idle-code groups into the SERDES Rx data stream. This ensures that the SERDES CTC has opportunities to rate-adapt for ppm offsets. The 2.5 Gbps Ethernet PCS IP core drives this signal high when the auto-negotiation process is running. At all other times, the IP core drives this signal low.
Management Signals		
mr_adv_ability[15:0]	In	**Advertised Ability** - Configuration status transmitted by PCS during auto-negotiation process.
mr_an_enable	In	**Auto-Negotiation Enable** - Active-high signal that enables auto-negotiation state machine to function.
mr_main_reset	In	**Main Reset** - Active-high signal that forces all PCS state machines to reset.
mr_restart_an	In	**Auto-Negotiation Restart** - Active-high signal that forces auto-negotiation process to restart.
mr_an_complete	Out	**Auto-Negotiation Complete** - Active-high signal that indicates that the auto-negotiation process is completed.
mr_lp_adv_ability[15:0]	Out	**Link Partner Advertised Ability** - Configuration status received from partner PCS entity during the auto-negotiation process. The bit definitions are the same as described above for the mr_adv_ability port.
mr_page_rx	Out	**Auto-Negotiation Page Received** - Active-high signal that asserts while the auto-negotiation state machine is in the “Complete_Acknowledge” state.
force_isolate	In	**Force PCS Isolate** – Active-high signal that isolates the PCS. When asserted, the Rx direction forces the GMII port to all zeros, regardless of the condition of the incoming 3.125 Gbps serial data stream. In the Tx direction, the condition of the incoming GMII port is ignored. The Tx PCS behaves as though the GMII Tx input port was forced to all zeros. Note, however, that the isolate function does not produce any electrical isolation – such as tri-stating of the GMII Rx outputs of the IP core. When the signal is deasserted (low), the PCS isolation functions are deactivated. The use of this signal is optional. If the user chooses not to use the isolate function, then this signal should be tied low.

Table 2-3. 2.5 Gbps Ethernet PCS IP Core Input and Output Signals (Continued)

Signal Name	I/O	Signal Description
force_loopback	In	**Force PCS Loopback** – Active-high signal that activates the PCS loopback function. When asserted, the 8-bit code-group output of the transmit state machine is looped back to the 8-bit code-group input of the receive state machine. When deasserted, the loopback function is deactivated. The use of this signal is optional. If the user chooses not to use the loopback function, then this signal should be tied low.
force_unidir	In	**Force PCS Unidirectional Mode** – Active-high signal that activates the PCS unidirectional mode. When asserted, the transmit state machine path between the Tx GMII input and the Tx 8-bit code-group output will remain operational, regardless of what happens on the Rx data path. (Normally Rx loss of sync, invalid code-group reception, auto-negotiation restarts can force the transmit state machine to temporarily ignore inputs from the Tx GMII port). When deasserted, the unidirectional mode is deactivated. The use of this signal is optional. If the user chooses not to use the unidirectional function, then this signal should be tied low.
an_link_ok	Out	**Auto-Negotiation Link Status OK** – Active-high signal that indicates that the link is OK. The signal is driven by the auto-negotiation state machine. When auto-negotiation is enabled, the signal asserts when the state machine is in the LINK_OK state. If auto-negotiation is disabled, the signal asserts when the state machine is in the AN_DISABLE_LINK_OK state (see IEEE 802.3 figure 37-6). This signal is intended to be used to produce the “Link Status” signal as required by IEEE 802.3, Status Register 1, Bit D2 (see IEEE 802.3 paragraph 22.2.4.2.13)
Miscellaneous Signals		
rst_n	In	**Reset** - Active-low global reset.
debug_link_timer_short	In	**Debug Link Timer Mode** – Active-high signal that forces the auto-negotiation link timer to run much faster than normal. This mode is provided for debug purposes (e.g., allowing simulations to run through the auto-negotiation process much faster than normal).

Parameter Settings

This IP core does not have any optional parameters.

IP Core Generation

This chapter provides information on how to generate the 2.5 Gbps Ethernet PCS IP core using the IPexpress tool in the Diamond software, and how to include the core in a top-level design.

Licensing the IP Core

An IP core and device-specific license is required to enable full, unrestricted use of the 2.5 Gbps Ethernet PCS IP core in a complete, top-level design. Instructions on how to obtain licenses for Lattice IP cores are given at:

http://www.latticesemi.com/products/intellectualproperty/aboutip/isplevercoreonlinepurchas.cfm

Users may download and generate the 2.5 Gbps Ethernet PCS IP core and fully evaluate the core through functional simulation and implementation (synthesis, map, place and route) without an IP license. The 2.5 Gbps Ethernet PCS IP core also supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited time (approximately four hours) without requiring an IP license. See "Hardware Evaluation" on page 17 for further details. However, a license is required to enable timing simulation, to open the design in the Diamond EPIC tool, and to generate bitstreams that do not include the hardware evaluation timeout limitation.

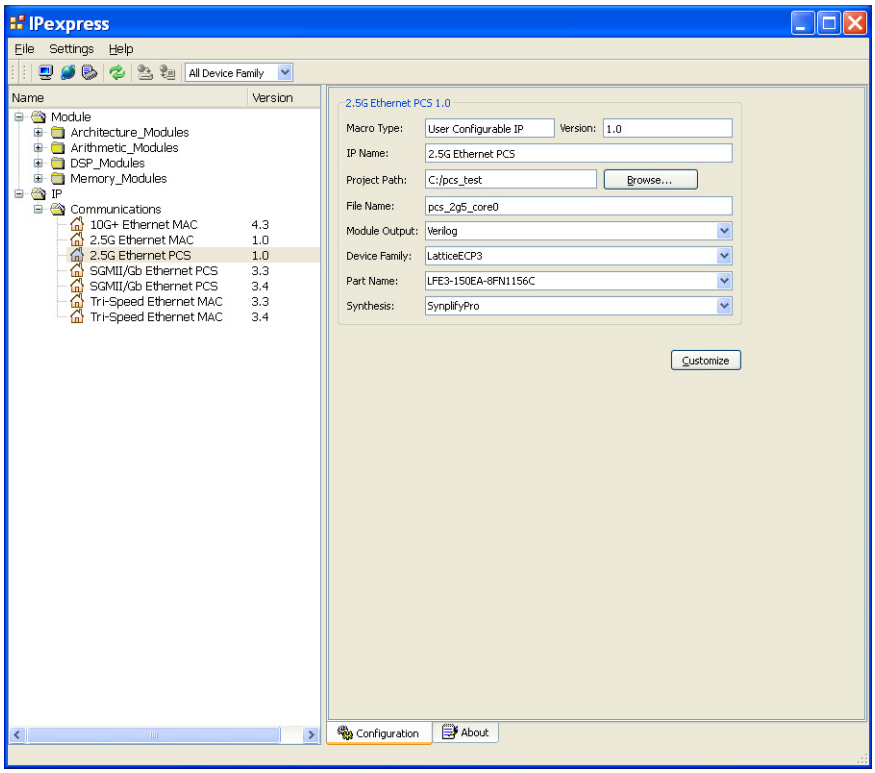
Getting Started

The 2.5 Gbps Ethernet PCS IP core is available for download from the Lattice IP server using the IPexpress tool. The IP files are automatically installed using ispUPDATE technology in any user-specified directory. After the IP core has been installed, the IP core will be available in the IPexpress GUI dialog box shown in Figure 4-1.

The IPexpress tool GUI dialog box for the 2.5 Gbps Ethernet PCS IP core is shown in Figure 4-1. To generate a specific IP core configuration the user specifies:

- **Project Path** – Path to the directory where the generated IP files will be located.
- **File Name** – "username" designation given to the generated IP core and corresponding folders and files.
- **Module Output** – Verilog or VHDL.
- **Device Family** – Device family to which the IP core is to be targeted. Only families that support the particular IP core are listed.
- **Part Name** – Specific targeted part within the selected device family.

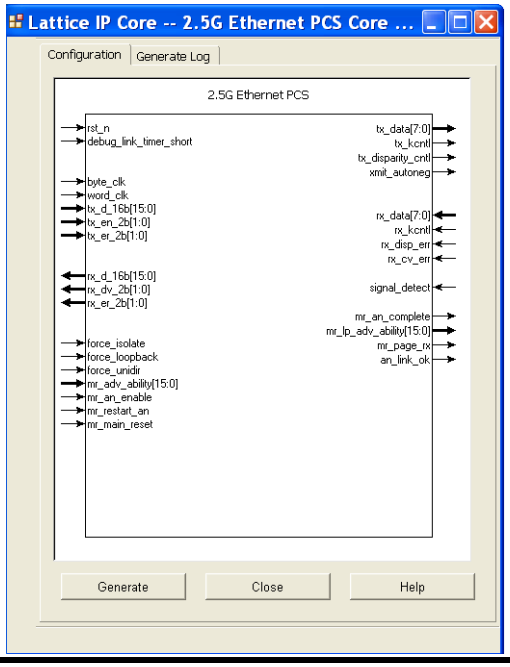
Figure 4-1. IPexpress Dialog Box



Note that if the IPexpress tool is called from within an existing project, Project Path, Module Output, Device Family and Part Name default to the specified project parameters. Refer to the IPexpress tool online help for further information.

To create a custom configuration, the user clicks the **Customize** button in the IPexpress tool dialog box to display the 2.5 Gbps Ethernet PCS IP core Configuration GUI, as shown in Figure 4-2.

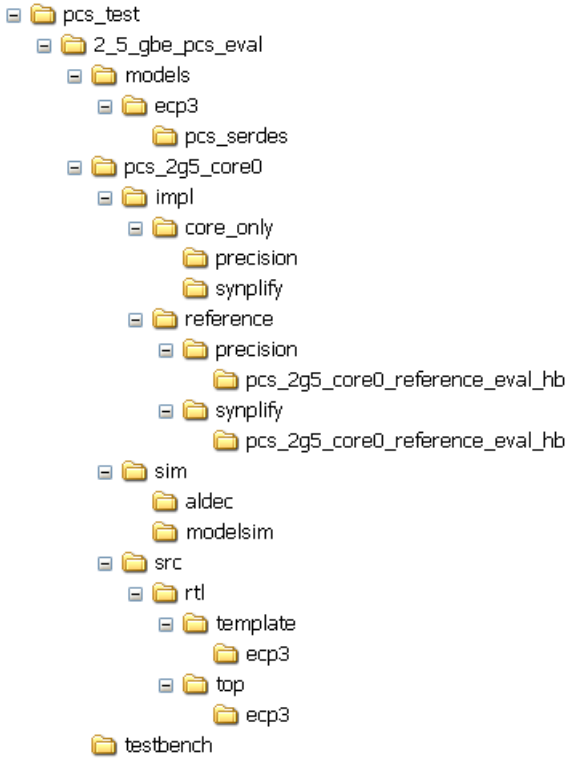
Figure 4-2. Configuration Dialog Box



IPexpress-Created Files and Top Level Directory Structure

When the user clicks the **Generate** button, the IP core and supporting files are generated in the specified "Project Path" directory. The directory structure of the generated files is shown in Figure 4-3.

Figure 4-3. 2.5 Gbps Ethernet PCS IP Core Generated Directory Structure



The design flow for IP created with the IPexpress tool uses a post-synthesized module (NGO) for synthesis and a protected model for simulation. The post-synthesized module is customized and created during the IPexpress tool generation.

Table 4-1 provides a list of key files created by the IPexpress tool. The names of most of the created files are customized to the user's module name specified in the IPexpress tool. The files shown in Table 4-1 are all of the files necessary to implement and verify the 2.5 Gbps Ethernet PCS IP core in a top-level design.

Table 4-1. File List

File	Description
<*username*>_inst.v	This file provides an instance template for the IP.
<*username*>_bb.v	This file provides the synthesis black box for the user's synthesis.
<*username*>_beh.v	This file provides a behavioral simulation model for the IP core.
<*username*>.ngo	This file provides the synthesized IP core.
<*username*>.lpc	This file contains the IPexpress tool options used to recreate or modify the core in the IPexpress tool.
<*username*>.ipx	IPexpress package file. This is a container that holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing this file to the associated Diamond project.

The following additional files providing IP core generation status information are also generated in the “Project Path” directory:

- **<username>_generate.log** – Diamond synthesis and map log file.
- **<username>_gen.log** – IPexpress IP generation log file.

The \<2_5_gbe_pcs_eval> and subtending directories shown in Figure 4-3 provide files supporting 2.5 Gbps Ethernet PCS core evaluation. The \<2_5_gbe_pcs_eval> directory contains files/folders with content that is constant for all configurations of the 2.5 Gbps Ethernet PCS. The \<*username*> subfolder (\pcs_2g5_core0 in this example) contains files/folders with content specific to the username configuration.

The \2_5_gbe_pcs_eval directory is created by IPexpress the first time the core is generated and updated each time the core is regenerated. A \<username> directory is created by IPexpress each time the core is generated and regenerated each time the core with the same file name is regenerated. A separate \<*username*> directory is generated for cores with different names, e.g. \<my_core_0>, \<my_core_1>, etc.

Instantiating the Core

The generated 2.5 Gbps Ethernet PCS IP core package includes black box (<username>_bb.v) and instance (<username>_inst.v) templates that can be used to instantiate the core in a top-level design. An example RTL top-level reference source file is provided in \<*project_dir*>\2_5_gbe_pcs_eval\<*username*>\src\rtl\top\<technology>.

The top-level file “top.v” implements the 2.5 Gbps Ethernet PHY reference design described in “Application Support” on page 18. Verilog source files associated with the reference design are located in the following directory: \<*project_dir*>\2_5_gbe_pcs_eval\<*username*>\src\rtl\template\<technology>.

The top-level file top_pcs_core_only.v supports the ability to implement just the 2.5 Gbps Ethernet PCS core by itself. This design is intended only to provide an indication of the device utilization associated with the 2.5 Gbps Ethernet PCS IP core; and it should not be used as useful example of a design application.

Using the SERDES with the 2.5 Gbps Ethernet PCS IP Core

Note that most applications for the 2.5 Gbps Ethernet PCS IP core require use of the FPGA SERDES block. The reference design demonstrates how the SERDES is used with the 2.5 Gbps Ethernet PCS IP core. However, note that the reference design only demonstrates one implementation – a single SERDES, assigned to channel 0. If your application requires a different SERDES configuration, for example utilizing a different channel number, or utilizing multiple channels, then you cannot use the SERDES module provided in the reference design. Instead, you must generate an appropriate SERDES module with the configuration settings required for your application. The SERDES module can be generated using IPexpress. Please see TN1176, *LatticeECP3 SERDES/PCS Usage Guide* for details about configuring the SERDES.

The following are a few key SERDES settings that must be chosen:

- Channel Protocol: GIGE
- Max Data Rate: 3.125 Gbps
- Reference Clock Rate: 312.5 MHz
- Tx/Rx Multiplier: 10X
- Tx/Rx Rate: Full
- FPGA Bus Width: 8
- CTC Block: ENABLED

Running Functional Simulation

The functional simulation model generated in the “Project Path” root directory (<*username*>_beh.v) may be instantiated in the user’s test bench for evaluation in the context of their design application. Lattice does not provide a test bench for evaluating this IP core in isolation. However, a functional simulation capability is provided in which <username>_beh.v is instantiated in the 2.5 Gbps PHY reference design described in “Application Support” on page 18. This FPGA top is instantiated in a test bench provided in `\<project_dir>\2_5_gbe_pcs_eval\testbench`.

Users may run the eval simulation by doing the following.

Using Aldec Active-HDL:

1. Open Active-HDL.
2. Under the Tools tab, select **Execute Macro**.
3. Browse to folder `\<project_dir>\2_5_gbe_pcs_eval\<username>\sim\aldec` and execute one of the "do" scripts shown.

Using Mentor Graphics ModelSim

1. Open ModelSim.
2. Under the File tab, select **Change Directory** and choose the folder `<project_dir>\2_5_gbe_pcs_eval\<username>\sim\modelsim`.
3. Under the Tools tab, select **Execute Macro** and execute the ModelSim “do” script shown.

The simulation waveform results will be displayed in the ModelSim Wave window.

Synthesizing and Implementing the Core in a Top-Level Design

The 2.5 Gbps Ethernet PCS IP core itself is synthesized and provided in NGO format when the core is generated through IPexpress. You may combine the core in your own top-level design by instantiating the core in your top-level file as described above in the “Instantiating the Core” section and then synthesizing the entire design with either Synplify or Precision RTL Synthesis.

The following text describes the evaluation implementation flow for Windows platforms. The flow for Linux and UNIX platforms is described in the Readme file included with the IP core.

As described previously, the top-level file top_pcs_core_only.v provided in `\<project_dir>\2_5_gbe_pcs_eval\<username>\src\rtl\top` supports the ability to implement the 2.5 Gbps Ethernet PCS core in isolation. Push-button implementation of this top level design is supported via the project file <*username*>_core_only_eval.ldf located in `\<project_dir>\2_5_gbe_pcs_eval\<username>\impl`.

To use this project file in Diamond:

1. Choose **File > Open > Project**.
2. Browse to `\<project_dir>\2_5_gbe_pcs_eval\<username>\impl\core_only\synplify (or precision)` in the Open Project dialog box.
3. Select and open <*username*>_core_only_eval.ldf. At this point, all of the files needed to support top-level synthesis and implementation will be imported to the project.
4. Select the **Process** tab in the left-hand GUI window.

5. Implement the complete design via the standard Diamond GUI flow.

Hardware Evaluation

The 2.5 Gbps Ethernet PCS IP core supports Lattice's IP hardware evaluation capability, which makes it possible to create versions of the IP core that operate in hardware for a limited period of time (approximately four hours) without requiring the purchase of an IP license. It may also be used to evaluate the core in hardware in user-defined designs.

Enabling Hardware Evaluation in Diamond

Choose **Project > Active Strategy > Translate Design Settings**. The hardware evaluation capability may be enabled/disabled in the Strategy dialog box. It is enabled by default.

Updating/Regenerating the IP Core

By regenerating an IP core with the IPexpress tool, you can modify any of its settings including: device type, design entry method, and any of the options specific to the IP core. Regenerating can be done to modify an existing IP core or to create a new but similar one.

Regenerating an IP Core in Diamond

To regenerate an IP core in Diamond:

1. In IPexpress, click the **Regenerate** button.
2. In the Regenerate view of IPexpress, choose the IPX source file of the module or IP you wish to regenerate.
3. IPexpress shows the current settings for the module or IP in the Source box. Make your new settings in the T**arget** box.
4. If you want to generate a new set of files in a new location, set the new location in the **IPX Target File** box. The base of the file name will be the base of all the new file names. The IPX Target File must end with an .ipx extension.
5. Click **Regenerate.** The module's dialog box opens showing the current option settings.
6. In the dialog box, choose the desired options. To get information about the options, click **Help**. Also, check the About tab in IPexpress for links to technical notes and user guides. IP may come with additional information. As the options change, the schematic diagram of the module changes to show the I/O and the device resources the module will need.
7. To import the module into your project, if it's not already there, select **Import IPX to Diamond Project** (not available in stand-alone mode).
8. Click **Generate**.
9. Check the Generate Log tab to check for warnings and error messages.
10. Click **Close**.

The IPexpress package file (.ipx) supported by Diamond holds references to all of the elements of the generated IP core required to support simulation, synthesis and implementation. The IP core may be included in a user's design by importing the .ipx file to the associated Diamond project. To change the option settings of a module or IP that is already in a design project, double-click the module's .ipx file in the File List view. This opens IPexpress and the module's dialog box showing the current option settings. Then go to step 6 above.

This chapter provides application support information for the 2.5 Gbps Ethernet PCS IP Core.

2.5 Gbps PHY Reference Design

This section describes the operation of a 2.5 Gbps PHY, using the Lattice 2.5 Gbps Ethernet PCS IP core. This design demonstrates how the IP core can be used in an Ethernet PHY application, and as a starting point for developing your own custom PHY design.

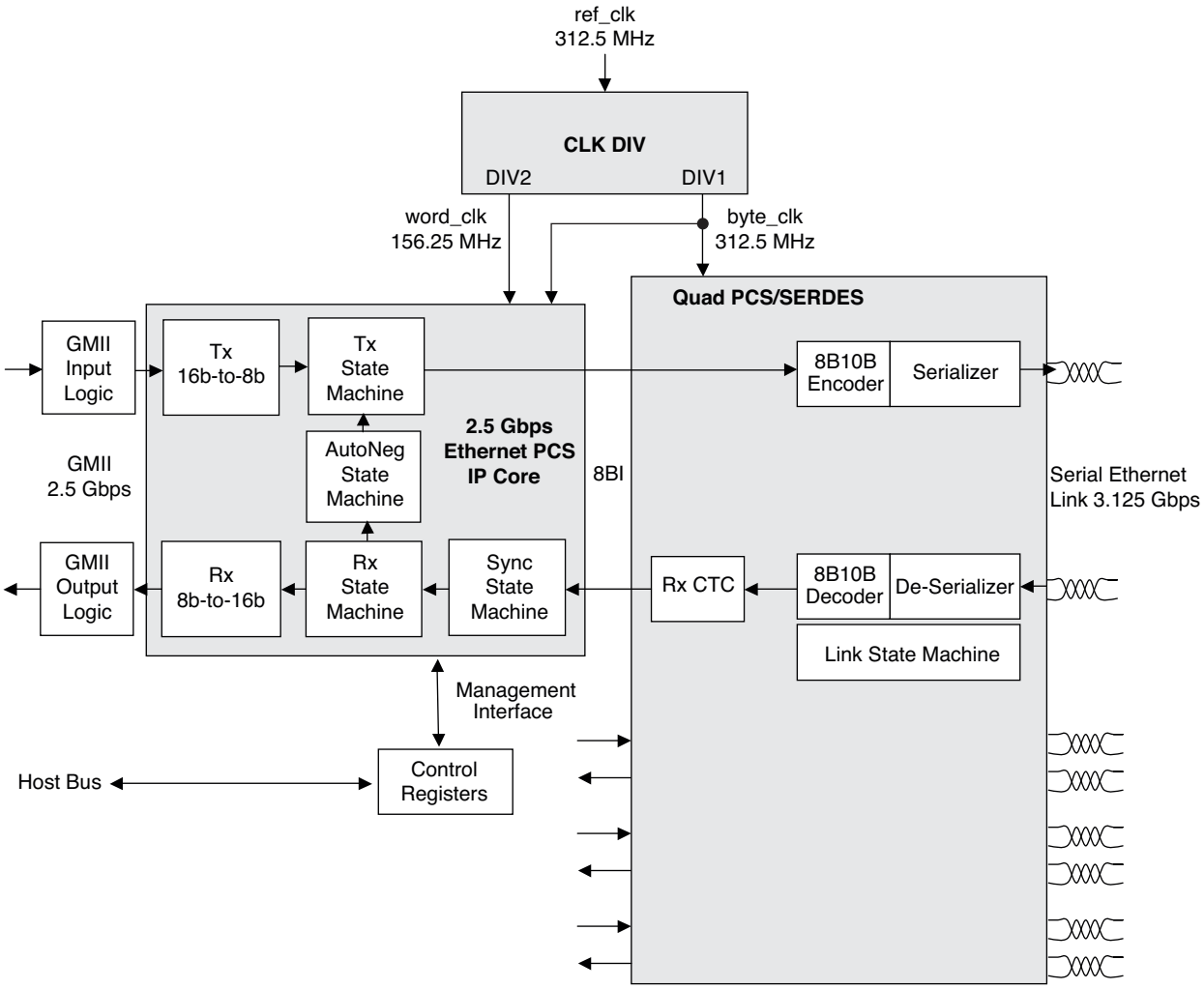
Features

- GMII Interface operating at 2.5 Gbps
- Differential, CML, Serial Port operating at 3.125 Gbps
- Management registers accessible through the Host Bus

Detailed Description

The block diagram of the 2.5 Gbps PHY is shown in Figure 5-1.

Figure 5-1. PHY with Host Bus Control Interface



The 2.5 Gbps Ethernet PCS IP Core

The IP core performs the data channel encoding/decoding, auto-negotiation, and byte/word gearing functions described earlier in the main section of this document.

PCS/SERDES

This block is an embedded circuit function within the FPGA architecture. It provides this application with a 3.125 Gbps SERDES function, a clock tolerance compensation circuit, and 8b10b data encoder/decoder functions.

GMII I/O Logic

The I/O logic consists of I/O flip-flops and buffers for moving GMII data into and out of the FPGA.

Control Registers

The control register block contains five of the management registers specified in IEEE 802.3, Clause 37 – Control, Status, Auto Negotiation Advertisement, Link Partner Ability, Auto Negotiation Expansion, and Extended Status. The register set is read/written through the parallel host-bus interface – the same parallel interface that the Lattice 2.5G MAC IP core uses to access 2.5G MAC registers.

The register map and register descriptions are shown below.

Table 5-1. Register Map

Address	Mode	Register Name
0x0	R/W	Control Register - Low
0x1	R/W	Control Register - High
0x2	R	Status Register - Low
0x3	R	Status Register - High
0x8	R/W	Advertised Ability - Low
0x9	R/W	Advertised Ability - High
0xA	R	Link Partner Ability - Low
0xB	R	Link Partner Ability - High
0xC	R	Auto Negotiation Expansion Register - Low
0xD	R	Auto Negotiation Expansion Register - High
0x1E	R	Extended Status Register - Low
0x1F	R	Extended Status Register - High

Table 5-2. Description of Control Register

Data Bit	Name	Mode	Description
Control Register High, Address 0x1 - Host Bus			
7	Reset	R/W	1 = Reset (self clearing), 0 = Normal
6	Loopback	R/W	1 = Loopback, 0 = Normal
5	Speed Selection[0]	Stuck-at-0	Combined with bit D6 in Control Register Low to form a 2-bit vector: Speed Selection [1:0] is stuck at “10” = 1Gbps
4	Auto Neg Enable	R/W	1=Enable, 0=Disable
3	Power Down	R/W	1=Power Down, 0=Power Up
2	Isolate	R/W	1=Isolate, 0=Normal
1	Restart Auto Neg	R/W	1=Restart (self clearing), 0=Normal
0	Duplex Mode	R/W	1=Full Duplex, 0=Half Duplex Note that the setting of this bit has no effect on the operation of the PCS channel. The PCS channel is always a 4-wire interface with separate Tx and Rx data paths.
Control Register Low Address 0x0 - Host Bus			
7	Collision Test	Stuck-at-0	1=Enable Test 0=Normal
6	Speed Selection[1]	Stuck-at-1	Combined with bit D5 in Control Register High to form the 2-bit vector Speed Selection [1:0]
5	Unidirectional	R/W	1=Unidirectional, 0=Normal
4	Reserved	Stuck-at-0	
3	Reserved	Stuck-at-0	
2	Reserved	Stuck-at-0	
1	Reserved	Stuck-at-0	
0	Reserved	Stuck-at-0	

Table 5-3. Description of Status Register

Data Bit	Name	Mode	Description
Status Register High Address 0x3 - Host Bus			
7	100BASE-T4	Stuck-at-0	0=not supported
6	100BASE-X Full Duplex	Stuck-at-0	0=not supported
5	100BASE-X Half Duplex	Stuck-at-0	0=not supported
4	10 Mbps Full Duplex	Stuck-at-0	0=not supported
3	10 Mbps Half Duplex	Stuck-at-0	0=not supported
2	100BASE-T2 Full Duplex	Stuck-at-0	0=not supported
1	100BASE-T2 Half Duplex	Stuck-at-0	0=not supported
0	Extended Status	Stuck-at-1	1=supported
Status Register Low Address 0x2 - Host Bus			
7	Unidirectional Capability	R	1=supported, 0=not supported
6	MF Preamble Suppress	Stuck-at-0	0=not supported
5	Auto Neg Complete	R	1=complete, 0=not complete
4	Remote Fault	Stuck-at-0	0=not supported
3	Auto Neg Ability	Stuck-at-1	1=supported
2	Link Status	R	1=Link Up, 0=Link Down Latch-on-zero Clear-on-read

Data Bit	Name	Mode	Description
1	Jabber Detect	Stuck-at-0	0=not supported
0	Extended Capability	Stuck-at-0	0=not supported

Table 5-4. Description of Advertised Ability Register

Data Bit	Mode	Name
Advertised Ability Register High, Address 0x9 - Host Bus		
7	R/W	Next Page
6	R/W	Acknowledge
5	R/W	Remote Fault[1]
4	R/W	Remote Fault[0]
3	R/W	0
2	R/W	0
1	R/W	0
0	R/W	Pause[1]
Advertised Ability Register Low, Address 0x8 - Host Bus		
7	R/W	Pause[0]
6	R/W	Half Duplex
5	R/W	Full Duplex
4	R/W	0
3	R/W	0
2	R/W	0
1	R/W	0
0	R/W	0

Table 5-5. Description of Link Partner Ability Register

Data Bit	Mode	Name
Link Partner Ability Register High, Address 0xB - Host Bus		
7	R	Next Page
6	R	Acknowledge
5	R	Remote Fault[1]
4	R	Remote Fault[0]
3	R	0
2	R	0
1	R	0
0	R	Pause[1]
Link Partner Ability Register Low, Address 0xA - Host Bus		
7	R	Pause[0]
6	R	Half Duplex
5	R	Full Duplex0
4	R	0
3	R	0
2	R	0
1	R	0
0	R	0

Table 5-6. Description of Auto Negotiation Expansion Register

Data Bit	Name	Mode	Description
Auto Negotiation Expansion Register High, Address 0xD - Host Bus			
7:0	Reserved	Stuck-at-0	Reserved
Auto Negotiation Expansion Register High, Address 0xC - Host Bus			
7:3	Reserved	Stuck-at-0	Reserved
2	Next Page Able	Stuck-at-0	0 = Not supported
1	Page Received	R	1 = received, 0 = Not received Latch on 1, clear on read
0	Reserved	Stuck-at-0	Reserved

Table 5-7. Description of Extended Status Register

Data Bit	Name	Mode	Description
Extended Status Register High, Address 0x1F - Host Bus			
7	1000BASE-X Full Duplex	Stuck-at-1	1 = Supported
6	1000BASE-X Half Duplex	Stuck-at-0	0 = Not supported
5	1000BASE-T Full Duplex	Stuck-at-0	0 = Not supported
4	1000BASE-T Half Duplex	Stuck-at-0	0 = Not supported
3:0	Reserved	Stuck-at-0	Reserved
Extended Status Register Low, Address 0x1E - Host Bus			
7:0	Reserved	Stuck-at-0	Reserved

Signal Descriptions

Table 5-8 shows a detailed listing of all the reference design I/O signals.

Table 5-8. Input and Output Signals for 2.5 Gbps PHY Reference Design

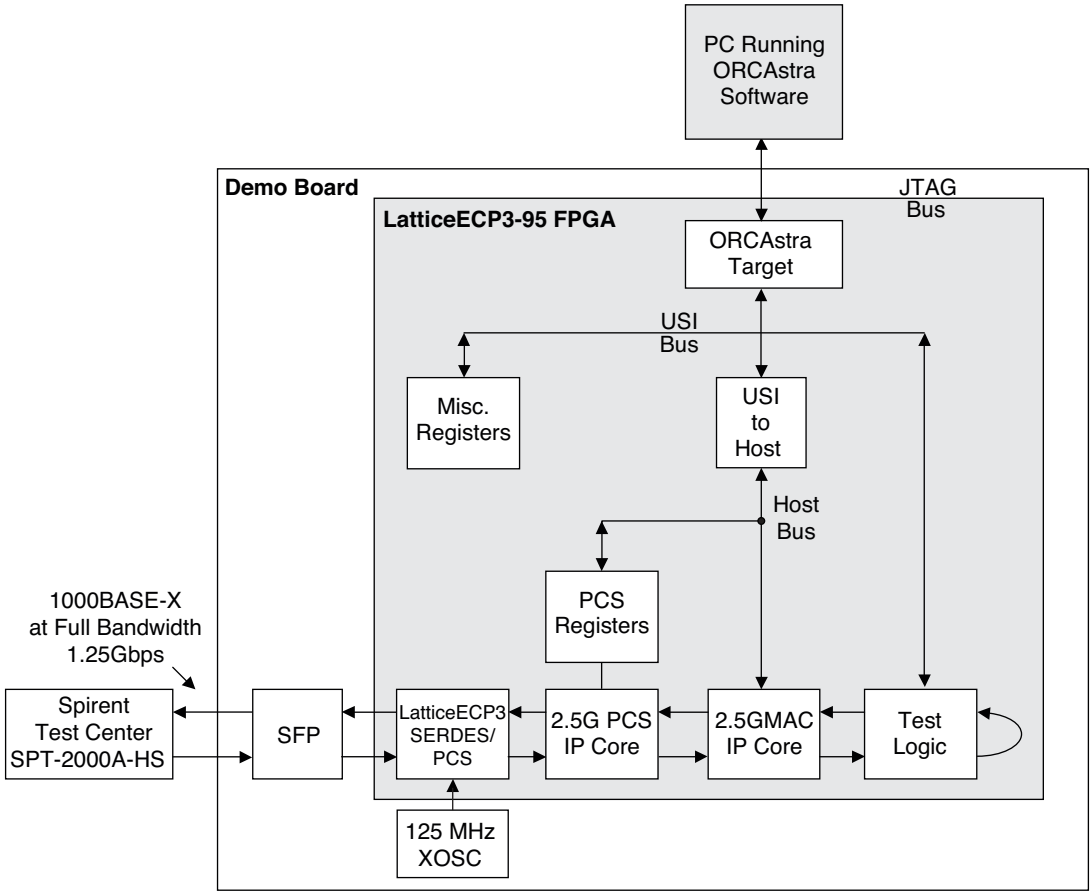
Signal Name	I/O	Description
125MHz Reference Clock Signals		
refclkp	In	**CML Reference Clock(P)**- P sense portion of differential SERDES reference clock (312.5 Mhz).
refclkn	In	**CML Reference Clock(N)**- N sense portion of differential SERDES reference clock (312.5 Mhz).
GMII Signals		
tx_d_16b[15:0]	In	**Transmit Data** - Incoming 16-bit GMII data. When the data frame reaches the Ethernet physical layer, the upper byte (bits 15:8) is sent first. Also, for each byte, the least significant bit is sent first (bit D8 for the upper byte; bit D0 for the lower byte).
tx_en_2b[1:0]	In	**Transmit Enable** - 2-bit, active-high signal that asserts when incoming GMII data is valid. tx_en_2b[1] is associated with the upper byte of the GMII data (bits 15:8). tx_en_2b[0] is associated with the lower byte of the GMII data (bits 7:0). Note that when an Ethernet frame is being transmitted, most of the time both transmit enable bits are asserted simultaneously. However at the beginning and end of the frame, it is possible for only one of the two GMII data bytes to be valid. A GMII data frame can begin and end on any byte (upper or lower).
tx_er_2b[1:0]	In	**Transmit Error** - 2-bit, active-high signal that denotes transmission errors -- or carrier extension events on the GMII Tx data port. tx_er_2b[1] is associated with the upper byte of the GMII data. tx_er_2b[0] is associated with the lower byte of the GMII data.
rx_d_16b[15:0]	Out	**Receive Data** - Outgoing 16-bit GMII data. When the data frame arrives at the Ethernet physical layer, the upper byte (bits 15:8) arrives first. Also, for each byte, the least significant bit arrives first (bit D8 for the upper byte; bit D0 for the lower byte)
rx_dv_2b[1:0]	Out	**Receive Data Valid** - 2-bit, active-high signal that asserts when outgoing GMII data is valid. rx_dv_2b[1] is associated with the upper byte of the GMII data (bits 15:8). rx_dv_2b[0] is associated with the lower byte of the GMII data (bits 7:0). Note that when an Ethernet frame is being received, most of the time both receive valid bits are asserted simultaneously. However at the beginning and end of the frame, it is possible for only one of the two GMII data bytes to be valid. A GMII data frame can begin and end on any byte (upper or lower).
rx_er_2b[1:0]	Out	**Receive Error** - 2-bit, active-high signal that denotes transmission errors -- or carrier extension events on the GMII Rx data port. rx_er_2b[1] is associated with the upper byte of the GMII data. rx_er_2b[0] is associated with the lower byte of the GMII data.
SERDES Signals		
HDOUTP0	Out	**Outbound Serial Data(P) Signal** - P-sense portion of differential SERDES transmit signal.
HDOUTN0	Out	**Outbound Serial Data(N) Signal** - N-sense portion of differential SERDES transmit signal.
HDINP0	In	**Inbound Serial Data(P) Signal** - P-sense portion of differential SERDES receive signal.
HDINN0	In	**Inbound Serial Data(N) Signal** - N-sense portion of differential SERDES receive signal.
Miscellaneous Signals		
rst_n	In	**Reset** - Active-low global reset.
mr_an_complete	Out	**Auto-Negotiation Complete** - Active-high signal that indicates that the auto-negotiation process is completed.
debug_link_timer_short	In	**Debug Link Timer Mode** – Active-high signal that forces the auto-negotiation link timer to run much faster than normal. This mode is provided for debug purposes (e.g., allowing simulations to run through the auto-negotiation process much faster than normal).
Management Interface Signals		
hclk	In	**Host Clock.** This is the host bus clock, and is used to clock the host bus interface.
hcs_n	In	**Host Chip Select.** This is an active-low signal used to select management registers for read/write operations.
hwrite_n	In	**Host Write.** This active-low signal is used to write data to the selected register.
haddr[5:0]	In	**Host Address.** This addresses one of the management registers.

Table 5-8. Input and Output Signals for 2.5 Gbps PHY Reference Design (Continued)

Signal Name	I/O	Description
hdatain[7:0]	In	**Host Data Input.** The CPU writes to the management registers through this data bus.
hdataout[7:0]	Out	**Host Data Output.** The CPU reads the management registers through this data bus.
hready_n	Out	**Host Ready. T**his is an active-low signal used to indicate the end of transfer. For write operations, hready_n is asserted after data is accepted (written). For read operations, hready_n is asserted when data on the hdataout bus is ready to be driven out to the CPU.

The 2.5 Gbps Ethernet PCS IP core has been validated using an Ethernet application design targeted to a LatticeECP3-95, -8 speed grade FPGA. The FPGA design included the 2.5G PCS IP core, the 2.5G MAC IP core, a data path loopback on the MAC client interface, and miscellaneous logic to control and read configuration/status registers. The FPGA was mounted on a demo board that included a 1000BASE-X SFP module. An optical cable was used to link the demo board to an external Spirent Testcenter Ethernet Analyzer. The system tests were performed at the 1 Gbps data rate. Figure 6-1 illustrates the setup.

Figure 6-1. 2.5G PCS Hardware Validation Setup



Chapter 7:

Support Resources

This chapter contains information about Lattice Technical Support, additional references, and document revision history.

Lattice Technical Support

There are a number of ways to receive technical support.

Online Forums

The first place to look is Lattice Forums (http://www.latticesemi.com/support/forums.cfm). Lattice Forums contain a wealth of knowledge and are actively monitored by Lattice Applications Engineers.

Telephone Support Hotline

Receive direct technical support for all Lattice products by calling Lattice Applications from 5:30 a.m. to 6 p.m. Pacific Time.

- For USA & Canada: 1-800-LATTICE (528-8423)
- For other locations: +1 503 268 8001

In Asia, call Lattice Applications from 8:30 a.m. to 5:30 p.m. Beijing Time (CST), +0800 UTC. Chinese and English language only.

- For Asia: +86 21 52989090

E-mail Support

- techsupport@latticesemi.com
- techsupport-asia@latticesemi.com

Local Support

Contact your nearest Lattice sales office.

Internet

www.latticesemi.com

References

The following documents provide more technical information regarding this IP core:

- IEEE 802.3-2002 Specification
- HB1009, *LatticeECP3 Family Handbook*
- TN1176, *LatticeECP3 SERDES/PCS Usage Guide*
- IPUG98, *2.5 Gbps Ethernet MAC IP Core User's Guide*

Revision History

Date	Document Version	IP Core Version	Change Summary
March 2012	01.0	1.0	Initial release.

Appendix A:

Resource Utilization

This appendix gives resource utilization information for Lattice FPGAs using the 2.5 Gbps Ethernet PCS IP core.

LatticeECP3 FPGAs

Table A-1. Performance and Resource Utilization[1]

Slices	LUTs	Registers	EBRs	External Pins	f_{MAX} (MHz)
550	600	900	0	106	312.5

1. Performance and utilization data are generated targeting an LFE3-150EA-8FN1156C device using Lattice Diamond 1.3 and Synplify Pro E-2011.03L software. Performance may vary when using a different software version or targeting a different device density or speed grade within the LatticeECP3 family.

Supplied Netlist Configurations

The Ordering Part Number (OPN) for the 2.5 Gbps Ethernet PCS core targeting LatticeECP3 devices is 2PT5GE-PCS-E3-U.